Digit Exp Math Educ (2016) 2:1–20
DOI 10.1007/s40751-015-0010-4

Enactivism, Spatial Reasoning and Coding

Krista Francis[1] · **Steven Khan**[2] · **Brent Davis**[1]

Published online: 2 December 2015

Abstract Drawing on an enactivist perspective in order to gain insight into how spatial reasoning develops and can be fostered, this article describes a study of how children engaged in spatial reasoning as they learned to program LEGO Mindstorms EV3 robots. Digital technologies afforded multiple opportunities for accumulating experiences for developing spatial reasoning that are difficult to come by in other contexts. Our video-recorded observations of children (aged 9 to 10) suggest that Bruner's enactive–iconic–symbolic typology of representations develop simultaneously rather than sequentially – the commonly held assumption. Furthermore, these same video observations provided insight into children's development of spatial reasoning through computer programming. Our findings have implications for how curriculum is designed and implemented in classrooms.

Keywords Enactivism · Computer programming · Robotics · Spatial reasoning · Video-analysis

[Krista Francis] I learned to code Fortran in the early 1980s without ever getting to run a program on the computer. The instructor marked the code. This was my only formal course in programming. As an articling actuary, I

✉ Krista Francis
kfrancis@ucalgary.ca

Steven Khan
skhan6@brocku.ca

Brent Davis
brent.davis@ucalgary.ca

[1] Werklund School of Education, University of Calgary, Calgary, AB, Canada

[2] Department of Teacher Education, Brock University, St. Catharines, ON, Canada

used a programming language (APL) for statistical analysis of large data sets on an IBM 486 PC that had a special APL microprocessor. Useful for manipulating arrays and matrices, APL had its own notation based on the Greek alphabet. We had a template that fit over the keys of the typewriter. Programming in APL meant learning an entirely different syntax that read from right to left. The great thing about programming with APL was that I could actually run the program to see if my code worked. It usually took an hour or two, sometimes overnight (Fig. 1).

The next language I learned was PHP/SQL for programming web pages. SQL stored data in tables and PHP called information from the database to display nicely on the screen. I struggled to learn how to program PHP/SQL because of the transitioning to object-oriented logic. Object-oriented was no longer sequential like Fortran or APL. With object-oriented code, a function or formula is defined and later invoked by a naming convention. Variables were passed along to the function when the name was invoked. Variables and functions or objects could be defined anywhere, including other files, and called into yours wherever, kind of like blocks of code. For instance, if I wanted a consistent header on every page of a large web site, I created a file called Header with the images and menu items I wanted to use. Then, for every page, I would just call the header and menu file to appear. This made changing the look of a site quite easy. Debugging the PHP code was challenging. One missed semi-colon and the entire program would not work. Finding the semi-colon often took longer than writing the program.

The importance of strong spatial reasoning skills for STEM disciplines has been well documented in the literature (see Benbow 2012; Casey et al. 2011; Wai et al. 2009). Most of what is known about spatial reasoning arises from a psychological perspective, one which is concerned with diagnostic tests (see Lehrer et al. 1998; Newcombe et al. 2013; Uttal et al. 2013). More recent investigations have sought insight into what spatial reasoning is, and how it develops, in order to foster spatial learning in educational settings (see Davis and the Spatial Reasoning Study Group 2015). Our study builds on this more recent understanding by describing how children engage in spatial reasoning as

Fig. 1 An APL keyboard (Rursus 2007)

they are using a laptop to program a LEGO Mindstorms EV3 robot[1] using its own software.

We first situate computer programing within an enactivist perspective. Next, we draw upon Bruner's enactive–iconic–symbolic typology of knowing for insight into the cognitive development of spatial reasoning. Then we describe programming in LOGO, Scratch and EV3 to describe how these child-friendly programming languages draw upon enactive–iconic–symbolic knowing and engage spatial reasoning. Following this, we discuss three video-recorded instances of children aged nine and ten programing with EV3 software.

Observations of children learning to program with this software do not appear to support the assumption that their representations develop sequentially from the experienced (enactive), through images and associations (iconic) to the purely abstract (symbolic). Furthermore, the same observations provided insight into the children's development of spatial reasoning through computer programming. Digital technologies afforded multiple opportunities for accumulating experiences to support the development of spatial reasoning differently from other contexts. Our search of the literature on coding in education turned up several examples of enactivism being used in relation to digital game design (see, for example, Ke 2014; Li 2012), but none in relation to computer programming.

Enactivism (in brief)

In our earlier work (Khan et al. 2015), grounded in the perspective of Varela et al. (1991), we have described enactivism as a theory of engagement that is simultaneously attentive to the coupling of organisms and their environments, to action as cognition and to sensori-motor co-ordinations. From an enactivist perspective, the environment plays a significant role in understanding the dynamic unfolding of cognitive processes: that is to say, the environment is always a (potential) *learning* environment in providing resources for thinking, for doing/knowing and for being. de Freitas and Sinclair (2014) argue that mathematical tools (e.g., an abacus for calculating or a compass for drawing a circle) "become parts of the learner, continually changing the very constitution of their bodies. […] Human bodies are constantly encountering, engaging and indeed amalgamating with other objects; the limits of our body are extended through these encounters" (p. 26).

Iseke-Barnes (1997) discusses such amalgamations when a body encounters and engages with a computer interface:

> As an example of the enactivist stance, consider the notion of computer interface. The interface is often considered the boundary between the user and the computer software. It may be considered to be the place where the two parties communicate or it may be considered a barrier that stands between the two, separating them. But from an enactivist stance, it is the place where the user and software co-emerge. The software becomes evident to the user through the interface. The interface is the means of occasioning the user's actions. The user takes action

[1] LEGO Mindstorms EV3 robots are a registered trademark of the Danish company LEGO.

> upon/within the software through the interface. The interface is thus the place where the software and the user codetermine each other. From the enactivist viewpoint, the medium and the user co-emerge. (pp. 62–63)

Further, viewed as an account of human learning, enactivist perspectives attend explicitly and deliberately to action, feedback and discernment. We have taken a position, one consistent with our enactivist framing, that spatial reasoning is both action *and* cognition, which we described as the constrained co-occurrence of sensory flux (sensation), recognition/discrimination (perception) and the situated movement of a body (or bodies) in the context of a goal-oriented situation (Khan et al. 2015).[2]

We suggest that the overt and visible forms of spatial reasoning observable as children assemble robots is complemented by other dimensions of spatial reasoning enacted by children as they program those robots. Most of the literature on coding/programming tends to be framed by more cognitivist sensibilities that focus mainly on the *symbolic manipulation* of code than the very notion of *manipulation* (i.e., literally, "*hand*-ling"), which already points to enactivist sensibilities about the role of sensori-motor co-ordination and feedback in the learning process.

Spatial Reasoning

Davis, Okamoto and Whiteley (2015, p. 141) describe the emergent complexity of spatial reasoning skills by means of a wheeled diagram (reproduced in Fig. 2 below). The wheel is intended to illustrate some of the critical elements of spatial reasoning, including the co-evolved and complementary nature of the mental and physical actions of spatial reasoning and the entangled and emergent nature of spatial skills. They comment how students "rapidly switch among various cognitive acts when drawing upon different forms of spatial data, just as they switch among representations when using spatial reasoning to solve problems [… Fig. 2] is a means to make sense of how spatial reasoning competencies arise, blend, and self transform" (p. 141).

Revisiting Bruner's Enactive–Iconic–Symbolic Sequence

To gain insight into how spatial reasoning develops, we next draw upon Bruner's enactive/iconic/symbolic sequence:

> We know and respond to recurrent regularities in our environment by skilled and patterned acts, by conventionalized spatioqualitative imagery and selective perceptual organization, and through linguistic encoding which, as so many writers have remarked, places a selective lattice between us and the physical environment. In short, the capacities that have been shaped by our evolution as tool users are the ones that we rely upon in the primary task of representation (Bruner 1964, p. 2).

[2] Readers who are interested in exploring enactivism more deeply might care to consult a recent special issue on this topic, *47*(2), of *ZDM*: *The International Journal of Mathematics Education*.

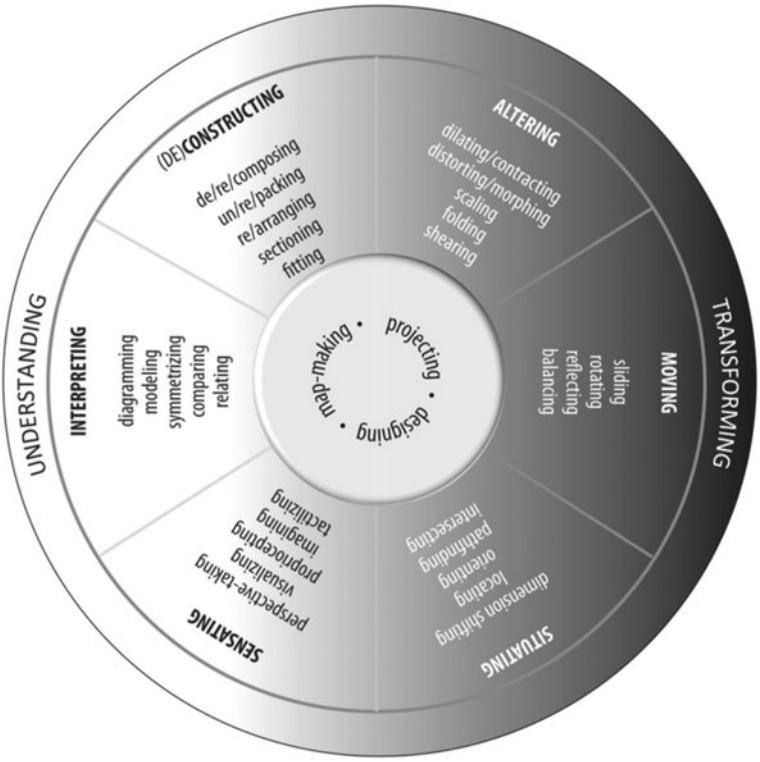

Fig. 2 Emergent complexity of spatial reasoning (used with permission)

By differentiation among enactive (action-based), iconic (image-based) and symbolic (language-based) experiences, Bruner proposed a development sequence through which learning was thought to progress. He described how higher-order representations depend on and arise in the combining of simpler components into an integrated task, illustrating his case by the example of a brain-injured man who could not recall the word "egg" when shown an egg. However, when given the opportunity to peel a cracked egg, this man could name it. He could only identify the object when it was encountered in a space of action. For Bruner, this incident exemplified an integration and internal dependency among enactive–iconic–symbolic representations, foregrounding in particular the necessity and primal importance of the enactive.

Bruner further claimed that representations developed in sequence: "Their appearance in the life of a child is in that order, each depending upon the previous one for its development" (p. 2). For instance, floors cannot be described without previously walking on them; the experiences are represented in our muscles (enactive representation). "Iconic representation summarizes events by the selective organization of percepts and of images, by the spatial, temporal and qualitative structures of the perceptual field and their transformed images" (p. 2). The image (iconic representation) stands for the perceptual event in the same way that a picture stands for an object. "Symbolic" refers to language, where a word stands for what it refers to.

As indicated by the linearity of the sequence, symbolic representation was considered the ultimate cognitive goal. Bruner saw young children's transitioning from iconic to symbolic as particularly concerning. As he viewed it, this transition is associated with both amplified cognitive possibility and an array of potential psychological problems, both of which arise as soon as children start to combine words and explore the effects of grammatical productiveness: "language provides a means, not only for representing experience, but also for transforming it. […] Translation of experience into symbolic form, with its attendant means of achieving remote reference, transformation, and combination, opens up realms of intellectual possibility that are orders of magnitude beyond the most powerful image forming system" (pp. 4, 13–14).

Importantly, Bruner recognized enactive, iconic and symbolic representations as mutually affecting:

> Let us propose that representation by such an action system [motor skills] is designed to guide and support symbolic activity. We believe that motoric representation for symbolic use makes possible the marvelous subtle articulatory side of language and lies at the base of the skills involved in technology. [… Once developed,] the three representational systems are parallel and each is unique, but all are also capable of partial translation into the other (Bruner et al. 1966, pp. 10–11).

The distinct characters and sequential nature of enactive, iconic and symbolic representations, as posited by Bruner a half-century ago, have been a focus in the mathematics education literature. For instance, Tall (2013) has highlighted how:

> Mathematical thinking begins in human sensorimotor perception and action and is developed through language and symbolism. […] This analysis is consonant with a combination of Bruner's enactive mode operating 'through action' and his iconic mode that involves not only visualisation but also depends upon visual or other sensory organization and upon the use of summarizing images (p. 11).

An implication of the unchallenged enactive–iconic–symbolic sequence is demonstrated in applications for developing instructional materials (see Buczynski et al. 2011; Muir and Cheek 1983; Winer and Schmid 1986). Mason (2008) argues that the sequence has become an established dogma for teaching mathematics, where learners "must begin with concrete manipulation (a somewhat startling metaphor when examined closely!) before being able to work with images, diagrams and thoughts, leading eventually to expressing and formalising in symbols. Unfortunately, one effect of this dogma is to contribute to a diminution of learners' powers to make mathematical sense" (p. 44) and may support formulaic teaching approaches.

Based on some of our recent observations of 8- and 9-year-olds, there is reason to believe that enactive, iconic and symbolic representations are neither distinct from one another nor necessarily developed sequentially in time. Rather, they are perhaps more productively understood as complexly emergent, co-occurring and co-dependent. Our findings are more consistent with Bruner's acknowledgement of the mutually affecting nature of enactive–iconic–symbolic representations.

To frame our discussion, we offer a nested rather than a linear graphic. One might portray the enactive, the iconic and the symbolic as "three nested circles, where each outer circle presupposes but transcends the ones within it" (Davis and Renert 2014, p. 87; see Fig. 3 below). In this image, the elliptical boundaries are intended to indicate permeable membranes and intertwining possibilities, rather than to suggest separation. The enactive representation is thus encapsulated within the iconic, and the enactive–iconic within the symbolic, reminding that the body is implicit/implicated in every cognitive act.

Shifts in Programming Using LOGO/Scratch/EV3

In this section, we describe three computer languages developed for children. First, we describe a program to draw a square in LOGO, a computer language that uses simplified syntax with a screen interface to provide instant feedback of a screen turtle following the programed instructions. Next, we describe a program to draw a square in Scratch. Similar to Logo, Scratch's two-dimensional screen interface provides instant feedback in the form of a cat following the programmed instructions. Then we describe an EV3 program that causes a robot to move through a square pattern.

LOGO

Papert (1980) developed the LOGO programming environment to put the child in control, where the "child programs the computer" (p. 19). LOGO uses language commands to direct a turtle icon to draw shapes on a computer screen.[3] The language is simplified compared with APL or Fortran, as described in the opening vignette, but it is still text-based: that is, in Bruner's terminology, LOGO programing falls in the space of symbolic representation, albeit more familiar than and much simplified from other computer languages.

Two examples of programs to draw a square are presented below. In the first case, on the left, the square is drawn through a sequence of discrete, ordered steps. In the second, the square emerges via a looping command that summons the same cluster of instructions four times.

```
TO SQUARE
FORWARD 200
RIGHT 90
FORWARD 200
RIGHT 90
FORWARD 200
RIGHT 90
FORWARD 200
RIGHT 90
END
```

```
TO SQUARE
REPEAT 4
FORWARD 200
RIGHT 90
END
```

[3] Note that the screen turtle icon evolved from an original turtle robot designed to carry out drawing functions (see Wikipedia https://en.wikipedia.org/wiki/Logo_(programming_language)).

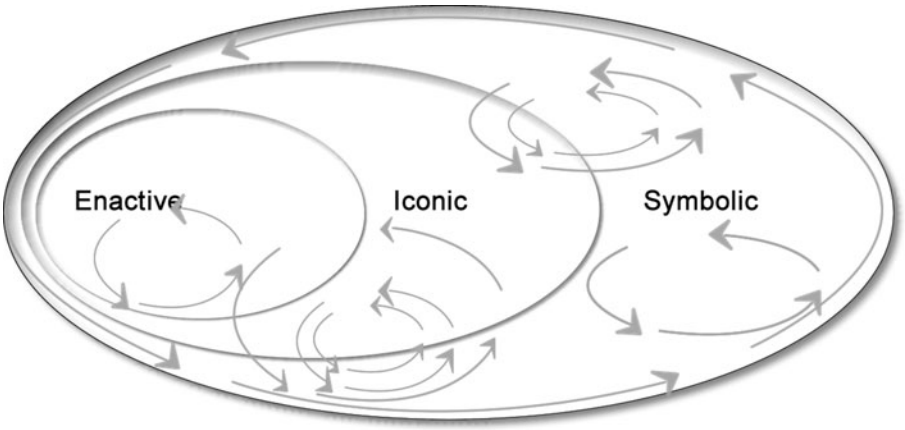

Fig. 3 Enactive, iconic and symbolic as nested, co-implicated and simultaneous

Figure 4 presents a screenshot of the turtle tracing a square with a LOGO interpreter. The interface provides instant feedback on the program.

The turtle in Fig. 4 follows a two-dimensional path on a flat computer screen. This requires the programmer to imagine and visualize the path that the turtle will follow, *unpacking*/*decomposing* the motion either into a series or into a repetition of equal forward movements and 90° rotations to the right. In doing so, the programmer must be able to transfer the key properties of a square (four equal sides and four right angles) into a path of forward moves and turns. Next, the programmer must add appropriate alphanumeric notation into the interface's programming box to map the two-dimensional motion of the turtle. In relation to Fig. 2's emergent wheel, the following aspects of spatial reasoning are identifiable in programming the turtle to trace a square: *imagining*, *visualizing*, *locating*, *path-finding*, *moving*, *rotating* and *decomposing*/*unpacking*.

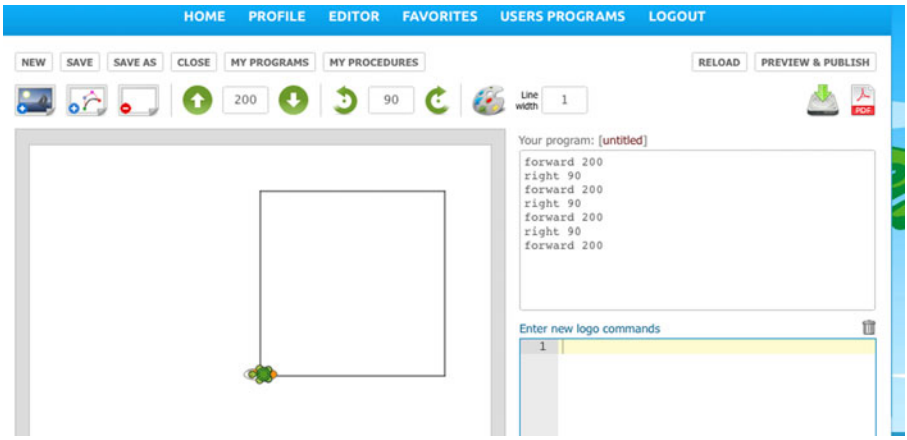

Fig. 4 Logo interface

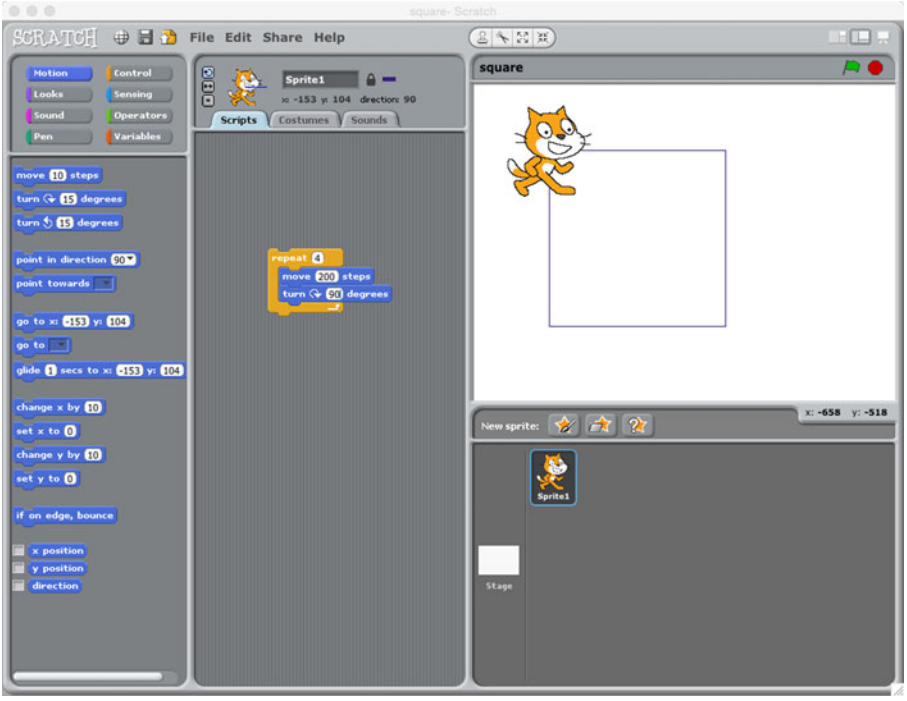

Fig. 5 Scratch code

Scratch

Scratch is a programming language that was developed for children by the MIT Media Lab's Lifelong Kindergarten group, led by Mitchel Resnick. Like LOGO, this programming language is available as a free download and is much simpler than other programming languages such as Fortran or Java.[4] Figure 5 below shows an image of a Scratch program for drawing a square.

Unlike with LOGO, in Scratch blocks of symbolic code are contained in iconic shapes. Colors indicate the classification of programming modes/actions. For instance, motion blocks are blue, logic/control blocks are gold, and pen blocks are green. Each block contains text-based commands, similar to the object-based programming found in PHP/SQL described in the opening vignette. The shapes fit together like a puzzle. Scratch's different shapes are intended to make elements of code easier to link together functionally.

These command-containing shapes are reminiscent of Tall's (2000) notion of *procept*: "the combination of [a] symbol representing both a process and the output of that process" (p. 36). Following Tall (2013), we would describe Scratch's encapsulated codes as *elementary* procepts: "a process that produces a mathematical object, and a symbol that is used to represent either process or object" (p. 45). The mathematical

[4] Scratch can also be interfaced with a robot (see MIT Media Lab https://www.media.mit.edu/sponsorship/getting-value/collaborations/wedo)

object is the color-coded shape containing the code, while the shape of the object represents how the code fits into the program.

Departing from the tidy distinctions offered in Bruner's typology, each elementary procept is an iconic–symbolic representation. Explicitly, it is encountered as an image, with important information presented (e.g., how it might connect to other members of its image set) by its very form. At the same time, its operation is highly symbolic, illustrated by the fact that it operates in a manner that prior programming languages call up through various word-commands.

The Scratch interface, like LOGO, enables the movement of a cat on a flat two-dimensional screen. As such, the spatial reasoning required for programming the cat to move in a square is the same as the spatial reasoning required to program the turtle as described above. Differently, however, Scratch requires fitting blocks of code together. In order to achieve this, the programmer must use the mouse to drag the specific and relevant code into the programming square and align it until it snaps together like puzzle pieces. This drag-and-drop and fitting together of the code requires the co-occurrence of spatial reasoning skills such as *moving*, *situating*, *sensating*, *interpreting* and *constructing*.

Additionally, both LOGO and Scratch can both be interfaced with robots. When a robot is added to the interface, the two-dimensional representation of the robots' movement in three-dimensions would occur simultaneously. When either LOGO or Scratch are interfaced with a robot, a two-dimensional screen serves as both the interface for programming and the two-dimensional representation of how the robot will "act" in three-dimensional space. These simultaneous two-dimensional and three-dimensional representations of the robot's movements likely support developing flexibility with *shifting* between *dimensions*.

EV3

The EV3 programming language is available as a free download, but it only works with costly Lego Mindstorms robots. Unlike LOGO and Scratch, the EV3 commands assembled on two-dimensional computer screen control the movement of the robot in three dimensions without a two-dimensional representation of the commands. With less text, EV3 is a more enactive–iconic language than either LOGO or Scratch. Figure 6 shows an EV3 program for a robot to trace out a square.

The arrow to the left in Fig. 6 indicates the start of the program. The green code block moves the motors (signified by the images of the motors, appearing at the left of those blocks). The learner can trigger the robot to move by dragging-and-dropping such coding blocks into the programming chain. Importantly, this can be accomplished with no formal knowledge of the symbolic meanings of the blocks. Feedback requires loading the program onto the robot in order to observe directly what actions (movements) the commands produce. The minimal syntax/symbolism (such as

Fig. 6 Screen capture of an EV3 program to follow a square path

numbers and special characters) on the coding blocks' lower tabs are used to fine tune the movement of the robot.

While both Logo and Scratch use simplified syntax to make programming easier for children, EV3 uses very little syntax at all. Each color-coded block represents chains of code/commands. These blocks contain the object-oriented code, similar to the PHP example of a header object described in the opening vignette. Following Tall (2013), we consider each coded block as a procept, which consists of "a collection of elementary procepts having the same object" (p. 45). The use of procepts for motion, sensing and logic in EV3 likely reduces the cognitive load of programming and renders programming more readily available for children.

Returning to Bruner's typology, with the simultaneity of the virtual world (where the programming happens) and the physical world (where the robot moves), we find ourselves unable to classify the elements of this sort of programming experience as singly enactive, iconic or symbolic. The activity is rooted in and organized around such *actions* as dragging and dropping on the screen; the experience is mediated by code-*images* that are strongly suggestive of their functions, and the actual outcomes are completely dependent upon inputting numbers and other *symbols* into the programing blocks. That is, EV3's procepts blend Bruner's notions of enactive, iconic and symbolic, in which the enactive–iconic combination appears most prominently in our observations.

All the forms of spatial reasoning used in programming with LOGO and Scratch are also utilized when programming with EV3: i.e., *imagining*, *visualizing*, *locating*, *path-finding*, *moving*, *rotating* and *decomposing*/*unpacking*. However, using a two-dimensional screen for programming a robot to move in three-dimensional space additionally requires different spatial thinking, i.e., *locating* and *orienting* in two dimensions, and then *locating*, *orienting* and *path-finding* in three dimensions. This shifting between dimensions adds considerably more complexity to the spatial reasoning required for programming a robot compared with moving an on-screen turtle or cat.

Description of the Research Context and Data Gathering

The research presented here was motivated in part by the thought that the categories of enactive, iconic and symbolic representations might be inadequate to describe children's sense-making as they learned to program in EV3. The data for this project were collected over five consecutive days during a robotics workshop, held during the early part of the school year in 2014. There were 18 child participants (12 girls and six boys), all between the ages of 9 and 10. Due to the timing and location away from the school of this study, parents were responsible for transporting their children to and from the workshop. Hence our sample is drawn from those children whose parents were sufficiently motivated to provide this opportunity for their children and were also able to manage these daily time commitments.

The 5-day developmental trajectory was based on an undergraduate engineering design competition format. For Day 1, the goals were for the children, first, to learn how to build the robot from the instruction manual and, second, to learn how to program the robot to move. On Day 2, the children learned how to program sensors for the robot to interact with the environment. On Day 3, the children were given the

challenge of removing diseased red trees from a LEGO-tree forest while leaving healthy green trees untouched. Day 4 involved collaborative work on designing and refining programs in order to complete the task as effectively as possible. Day 5 was a group-based competition where each team had two attempts at the challenge.

Students were video- and audio-recorded for the duration of the workshop by a professional videographer who moved among the groups of students. The video allows the viewer to slow down the process and identify integrated/nested processes of spatial (and other) activity that were occurring. All participants (and their parents) gave consent to be video-recorded during the workshop. The videographer was instructed to focus on trying to capture gestures, expressions, embodied actions, discussions, interactions and on-screen programming.

Data Analysis

From the video data corpus, we have selected three brief segments that afford evidence of bodily engagement in spatial reasoning while coding. To fit our descriptions to the video-clips (which are linked to the text), our descriptions are offered in the present tense. In the first such clip, we observe Jayda swinging her body and feet in correspondence with dragging-and-dropping coding blocks into the program. In the second extract, Liam is observed swaying right then left as he swipes his fingers right then left along the trackpad. Later in the same clip, Jared slides his tongue between his lips as his fingers swipe along the trackpad. In the third and final video clip, Jared gestures with his hands to describe what the programming code means when it turns the robot.

Clip 1 – Jayda Coding Using her Feet[5]

Jayda (*Jayda coding using her feet* (https://vimeo.com/143799690)) is programming with the EV3 software by means of moving program blocks onto the screen. She begins pushing the chair back slightly before she swings her feet forward. As her feet swing back, she curls her legs up beneath her, and her torso moves forward toward the screen and her fingers move along the track pad. As she scrolls, her fingers and her feet move backward as her body moves forward, bringing the chair closer. She moves closer to the screen as the block approaches the chain (Fig. 7).

Jayda pushes and straightens her fingers away from her, then straightens her body away from the screen, pushing her chair away slightly. She lifts her fingers to reposition them at the bottom of the trackpad and moves her feet forward slightly. Her whole body then pushes forward as she scrolls up to move a block onto the programming line (Fig. 8).

Jayda's suite of movements, in tandem with manipulating objects on the computer screen, is representative of an enactivist framing, following de Freitas and Sinclair's (2014) notion of amalgamation with interface and Iseke-Barnes' (1997) notion of co-emergence of the user and the software. As Jayda took action on the EV3 software through the interface, she and the software co-evolved through the interplay between the computer screen and the touchpad. To find the right block of code, she *compared*

[5] See the video *Jayda coding using her feet* at: https://vimeo.com/143799690

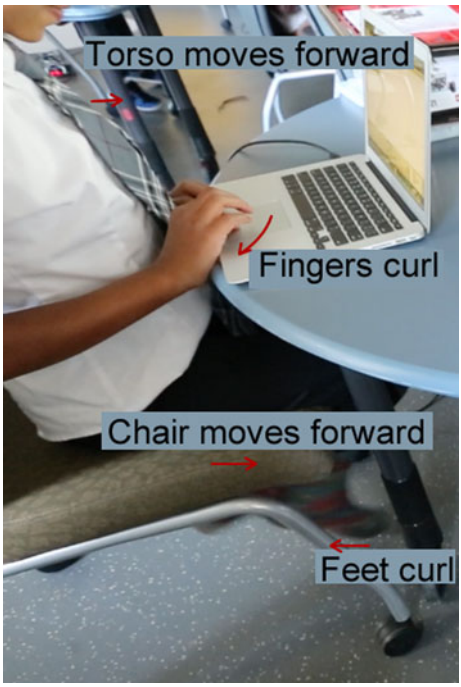

Fig. 7 Jayda selecting a program block

Fig. 8 Dropping a program block on the chain

the procepts to find the one related to movement. Once selected, she dragged (*moved*) the procept to the programming chain. She aligned (*located*, *oriented*) the procept before dropping it (*fitting*) onto the chain, all the while *imagining*/ *visualizing* the robot's intended movement (*pathfinding*) and *feeling*/holding the mouse. In this instance, Jayda's bodily actions and engagement, the EV3 software and the laptop were all unified. Following Davis, Okamoto and Whiteley (2015), the broader categories of spatial reasoning occurring simultaneously were *sensating*, *interpreting*, *constructing*, *moving* and *situating*.

Clip 2 – Boys Coding Using Trackpad[6]

In this clip (*Boys coding using trackpad* (https://vimeo.com/143801371)), we cannot see the screen. The video-record comes from the first day of the camp, and was taken as Liam and Jared learn how to program the robot to move. To achieve this, blocks are dragged and dropped onto the chain of code. Liam swings his whole body in conjunction with sliding his fingers along the trackpad. First, he slides his body to the right, just before he slides his fingers to the right along the trackpad. Liam repeats the process as he slides his body and fingers to the left. His whole body is engaged as he slides programming blocks on the screen (Fig. 9).

Liam's repeated sliding of his chair, body and fingers in both directions functions is a sign for Jared that something is not working. If Liam were to pick up a block of code and drag it into the chain, he would only need to move the block in one direction. Unable to see Liam's screen, Jared perhaps notices and responds to the bodily swaying back and forth, together with the trackpad motions, as an indication that Liam is having difficulty.

Jared moves over to assist Liam. As Jared uses the trackpad, he engages his tongue in the action, which protrudes when his fingers touch the trackpad, an action consistent with selecting a code block. Then his tongue retracts as his fingers slide upwards, an action consistent with sliding the block toward the chain. A few seconds later, he slides his tongue to the right as his fingers slide to the right on the trackpad, consistent with sliding the block to the right on the screen. His releases his tongue only when the block is in place on the programming chain (Fig. 10).

On our initial analysis, the simultaneous mirrored nature of Liam's bodily actions and his spatial reasoning was not noticed. Had we seen Liam's bodily actions in a classroom setting – that is, solely in the actual moment of engagement – we likely would have interpreted them simply as extraneous and irrelevant fidgeting. This initial oversight highlights the subtlety of bodily enactments of spatial reasoning and the strength of video data for capturing them and permitting slowed-motion observations to be made.

What is evident in these two video clips is spatial reasoning (as we have defined it) as both action *and* cognition. Spanning Bruner's typology, it appears to be enactive–symbolic; that is, involving both action (observable, body-engaged) and cognition (not-directly-observable) during the constrained co-occurrence of sensory flux (sensation), recognition/discrimination (perception) and situated movement of a body (or bodies) in the context of a goal-oriented situation.

[6] See the video *Boys coding using trackpad* at: https://vimeo.com/143801371

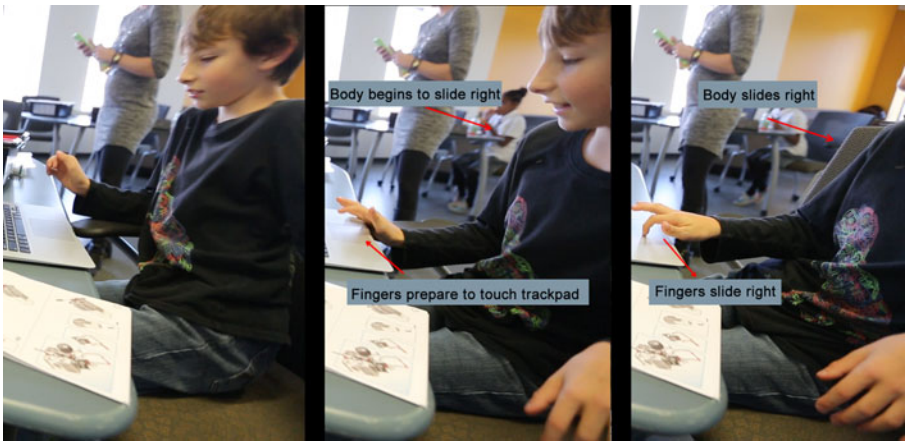

Fig. 9 Liam sliding his body as he slides his fingers along the trackpad

Elsewhere (Khan et al. 2015), we have argued that enactivism provides an appropriate framework for studying spatial reasoning. This theory of knowing and learning "is concerned with *learning in action* since it is the potential for action in the world that focuses attention and drives learning" (p. 272; *italics in original*). In the above scenarios, the actions unfold across several coupled and interacting worlds simultaneously – viz, the on-screen coding environment in which blocks of code have to be manipulated, the relationship of these on-screen icons to the desired future movements of the robot in the physical world, the actions of the learners' physical bodies and the social domains of those actions. From the wheel image depicted in Fig. 2, the spatial skills needed for this on-screen coding include *interpreting* the meaning of the coding blocks in terms of *situating* the robot's actions, as well as *sensating* the mouse and the screen in order to *move* and *situate* the blocks of code in the *construction* of the programming chain. If learning is to be understood in terms of on-going construal by which one both elaborates possibilities in and maintains coherence with such multiple domains (as were identified in the previous sentence), then it seems unlikely that

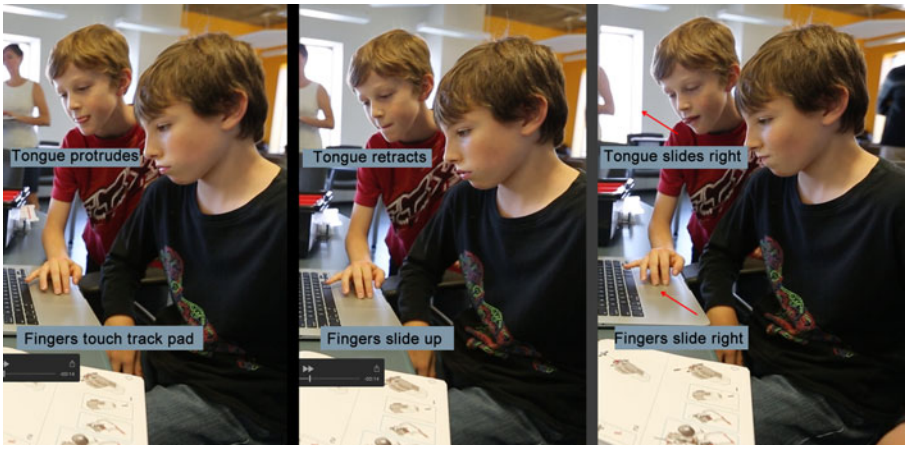

Fig. 10 Jared helping Liam

learning sequences might be adequately characterized as linear and directed sequential movement through enactive, iconic and symbolic modes of representation. That is not to suggest that the classification is not useful, however. Rather these three "modes" of enactive, iconic and symbolic are perhaps better construed as "nodes" in networks of nested, entangled and transcendent engagements.

Clip 3 – Coding for Wheel Rotations[7]

In Clip 3 (*Coding for wheel rotations* (https://vimeo.com/143802565)), the group is learning how to make a robot follow a line using a light/colour sensor. For the robot to follow a line, a sensor has to be used to input information on whether or not a particular colour is present. If the colour is present, the wheels rotate in a manner that moves the robot slightly forward, before it re-assesses the situation. If the colour is still present, then the robot repeats the action. If the colour is no longer present, the wheels rotate slightly in the other direction before re-assessing, in the hope of re-locating the colour. The net effect of this forward-and-backward action is a narrow zigzag (/\/\/\) as the robot tracks forward along the line.

Learning to program the robot to perform such forward and backward movements can be tricky, as the interface and commands in EV3 measure distances in wheel turns (rather than centimeters or inches) and angles in relative wheel turns (rather than degrees). So, for instance, spinning one wheel backward a half-turn and the other wheel forward a half-turn will rotate the robot by about 80°. Hence, for children just learning to program a robot's movement, a common misconception is to assume a direct one-to-one correspondence between the slider power value and the degree measure of a turn – for example, believing that setting the slider to 78 % will turn the robot through 78° (see Fig. 11 below). The relationship between slider value and degree measure of turn is a proportional one, and this is a more challenging concept than a direct measure. The complexity of the learning situation is increased as a change in direction (amount of rotation) is a function of the relationship between the power difference applied to the left and right motors.

In Clip 3, the group is discussing their issues with getting the robot to follow the line. One girl states that she "turned it 78° and it still did not work." Noticing that the girl thinks the robot should turn 78°, Jared explains to the videographer that "78° is how many times the wheels turn". In his explanation, he circles his hands around each other (see Fig. 12 below). This motion is consistent with the circular motion of the motors, but not the rotation of the robot. Jared understands that the (symbolic) number relates to the motors. He does not appear to understand that the slider relates to the percentage of power directed to them.

The interface is visually simple but conceptually dense; its relationship to actions in the physical world entails working with multiple pieces of information. However, because of the ease of changing aspects without having to work with symbolic code, coupled with the rapid feedback from the robotic actions that coincide with the coding commands, learners are able to progress relatively quickly toward understanding and interpretations that are sufficient to accomplish the assigned task (in this case, making the robot move along a line). Even without knowing exactly what the symbols on the

[7] See the video *Coding for wheel rotations* at: https://vimeo.com/143802565

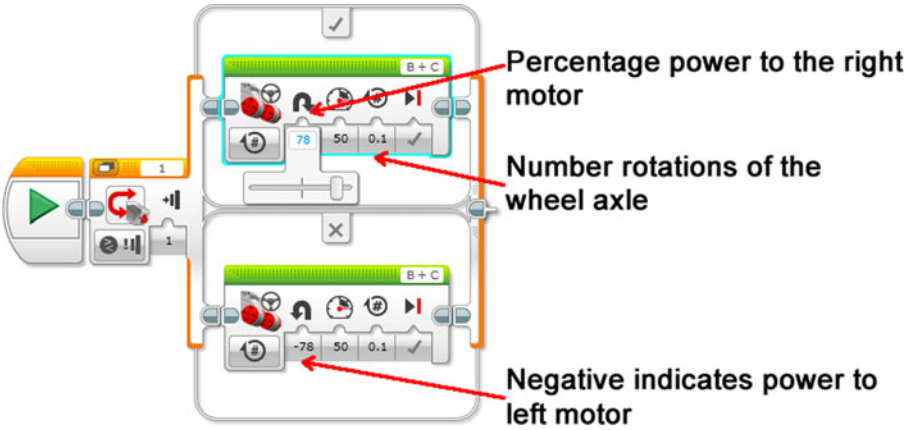

Fig. 11 Program to turn a robot

visual on-screen interface mean, the learner is able to engage in appropriate action that is directed towards a specific goal.

Discussion

Enactivism focuses on learning *in* action – as opposed to learning *from* action, which is better aligned with other embodied approaches. For us, the brief episodes described above of children programming robots to sense-and-move-through their environments exemplify learning *in* action. The affordances of the associated video recordings – such as the opportunities to isolate individuals, to watch repeatedly, to slow motion (thereby enabling subtle correspondences among actions to be noticed) – permit observations

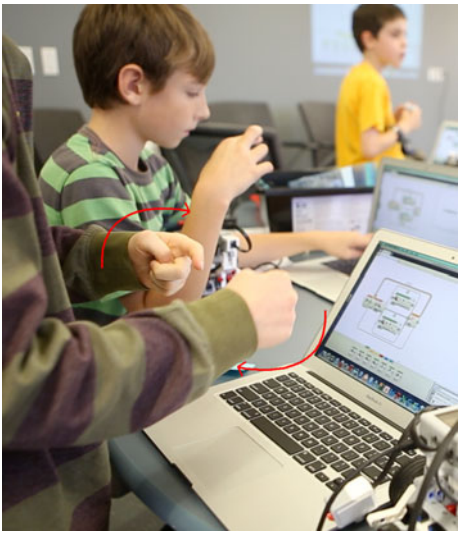

Fig. 12 Jared gestures to communicate his understanding of how the wheels turn

that might not otherwise be possible in a fast-moving classroom setting involving so many actors and factors.

In particular, by slowing the motion of the videos, one is able to notice the co-emergent engagement of many of the spatial reasoning skills itemized in the wheel image shown in Fig. 2. Imagining how the robots will move in three-dimensional space or composing computer programming code in two-dimensional space to test and observe how the robot will move in three-dimensional space require both sophisticated thinking as well as fluency in shifting between two and three dimensions. The interplay among the varied spatial skills highlights how educational tasks can provide possibilities for developing complex spatial reasoning.

For us, by far the most compelling affordance of an enactivist frame, coupled with the technology of video recording, is the challenge it presents to some of formal education's tidy separations and sequences. For instance, among the many actions that we have been able to analyze, we find it interesting that the ones that most occupied our attention are those subtle movements that are so easily dismissed (e.g., as fidgetings or mere distractions). The realization that consistently across many observations these movements co-occur with formal acts of spatial reasoning compels us to wonder whether the common requirement for children to sit still in their seats might limit the development of their spatial reasoning.

The simultaneity of the subtle bodily actions and formal on-screen iconic–symbolic manipulations also presents a challenge for us in regard to the associated curricular sequential dogma of Bruner's Enactive–Iconic–Symbolic typology, which, as Mason (2008) has observed, serves to legitimize the idea of always starting learners off with concrete enactive experiences, then proceeding to iconic representations, and only then moving to more abstract symbolic representations. If, however, these representations are simultaneous, co-entangled, nested and transcendent, then curricular experiences should likewise provide intertwined opportunities for cognitive development. In the design of the workshop, for example, learners worked across all of these representational modes continuously and simultaneously.

In discussions of curriculum planning, a frequent companion concern to the topic of sequencing is the matter of adequacy of practice. What constitutes sufficient experience or practice? This is a question that is not easily answered, in part because it varies so dramatically across learners. How, when and where do learners assemble sets of experiences that enable them to transcend thinking strategies that are tethered to particular actions and images, in order to reason abstractly about the workings of the world? In the instance of this study, what does the interplay of entangled worlds – bodily, robotic and virtual (the coding environment) – provide in terms of learners exploring sufficient and diverse sets of experiences of reasoning about the action in the physical world of a non-human agent (the robot) as mediated by a virtual world interface?

We could go on, but our central point has been made: emerging educational tools reveal that many of the orthodoxies (i.e., literally "true or straight opinions") of formal education are shown to be not just simplifications, but limitations. Overcoming such constraints, we suspect, will involve combining new means of interpreting, new foci for learning and new tasks for engagement. Such is the spirit in which we consider enactivism, spatial reasoning and coding.

Acknowledgments This research was funded by the Imperial Oil Science Engineering and Technology (IOSTEM) Education Initiative. We thank the teachers and children who participated in the IOSTEM Academy. We are also grateful to Michael Poscente for designing and coaching the LEGO robotics task.

References

Benbow, C. (2012). Identifying and nurturing future innovators in science, technology, engineering, and mathematics: a review of findings from the study of mathematically precocious youth. *Peabody Journal of Education, 87*(1), 16–25. doi:10.1080/0161956X.2012.642236.

Bruner, J. (1964). The course of cognitive growth. *American Psychologist, 19*(1), 1–15.

Bruner, J., Olver, R., Greenfield, P., Hornsby, J., Kenney, H., MacCoby, M., & Modiano, N. (1966). *Studies in cognitive growth*. New York: Wiley.

Buczynski, S., Gorsky, J., McGrath, L., & Myers, P. (2011). Sift like eratosthenes. *Teaching Children Mathematics, 18*(2), 110–118.

Casey, B., Dearing, E., Vasilyeva, M., Ganley, C., & Tine, M. (2011). Spatial and numerical predictors of measurement performance: the moderating effects of community income and gender. *Journal of Educational Psychology, 103*(2), 296–311. doi:10.1037/a0022516.

Davis, B., & Renert, M. (2014). *The math teachers know: Profound understanding of emergent mathematics*. New York: Routledge.

Davis, B., & the Spatial Reasoning Study Group. (2015). *Spatial reasoning in the early years: Principles, assertions, and speculations*. New York: Routledge.

Davis, B., Okamoto, Y., & Whiteley, W. (2015). Spatializing school mathematics. In B. Davis & the Spatial Reasoning Study Group (Eds.), *Spatial reasoning in the early years: Principles, assertions, and speculations* (pp. 139–150). New York: Routledge.

de Freitas, E., & Sinclair, N. (2014). *Mathematics and the body: Material entanglements in the classroom*. Cambridge: Cambridge University Press.

Iseke-Barnes, J. (1997). Enacting a chaos theory curriculum through computer interactions. *Journal of Computers in Mathematics and Science Teaching, 16*(1), 61–89.

Ke, F. (2014). An implementation of design-based learning through creating educational computer games: a case study on mathematics learning during design and computing. *Computers & Education, 73*, 26–39. doi:10.1016/j.compedu.2013.12.010.

Khan, S., Francis, K., & Davis, B. (2015). Accumulation of experience in a vast number of cases: Enactivism as a fit framework for the study of spatial reasoning in mathematics education. *ZDM: The International Journal of Mathematics Education, 47*(2), 269–279. doi:10.1007/s11858-014-0623-x.

Lehrer, R., Jenkins, M., & Osana, H. (1998). Longitudinal study of children's reasoning about space and geometry. In R. Lehrer & D. Chazan (Eds.), *Designing learning environments for developing understanding of geometry and space* (pp. 137–167). Mahwah: Lawrence Erlbaum.

Li, Q. (2012). Understanding enactivism: a study of affordances and constraints of engaging practicing teachers as digital game designers. *Educational Technology Research and Development, 60*(5), 785–806. doi:10.1007/s11423-012-9255-4.

Mason, J. (2008). Joined-up reflections: part 1. *Mathematics Teaching, 210*, 42–44.

Muir, S. & Cheek, H. (1983). *A developmental mapping program integrating geography and mathematics*. ERIC document: eric.ed.gov/id=ED238796.

Newcombe, N., Uttal, D. & Sauter, M. (2013). Spatial development. In P. Zelazo (Ed.), *Oxford handbook of developmental psychology: vol. 1 Body and mind* (pp. 564–590). New York, NY: Oxford University Press.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books.

Rursus (2007). An *APL keyboard* [Digital]. Retrieved from: http://commons.wikimedia.org/wiki/File:APL-keybd2.svg

Tall, D. (2000). Technology and versatile thinking in mathematics. In M. Thomas (Ed.), *Proceedings of the Technology in Mathematics Education* 2000 Conference (pp. 33–50). Auckland, NZ: Auckland

University of Technology. Retrieved from https://homepages.warwick.ac.uk/staff/David.Tall/pdfs/dot2000g-time2000.pdf

Tall, D. (2013). *How humans learn to think mathematically: Exploring the three worlds of mathematics*. New York: Cambridge University Press.

Uttal, D., Meadow, N., Tipton, E., Hand, L., Alden, A., Warren, C., & Newcombe, N. (2013). The malleability of spatial skills: a meta-analysis of training studies. *Psychological Bulletin, 139*(2), 352–402. doi:10.1037/a0028446.

Varela, F., Thompson, E., & Rosch, E. (1991). *The embodied mind: Cognitive science and human experience*. Cambridge: MIT Press.

Wai, J., Lubinski, D., & Benbow, C. (2009). Spatial ability for STEM domains: aligning over 50 years of cumulative psychological knowledge solidifies its importance. *Journal of Educational Psychology, 101*(4), 817–835.

Winer, L., & Schmid, R. (1986). Using Brunerian learning theory with educational simulations to teach concepts. *Canadian Journal of Educational Communication, 15*(3), 153–165.